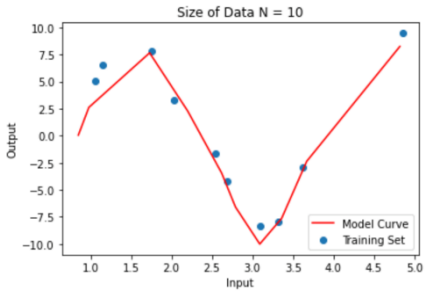
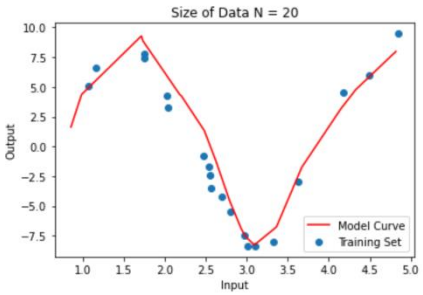
# CS5691 Assignment 2

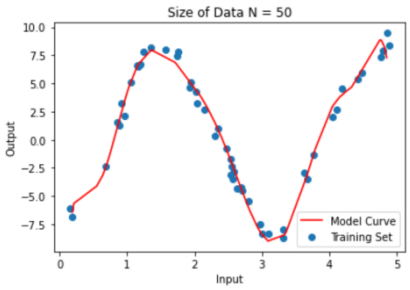
## Regression

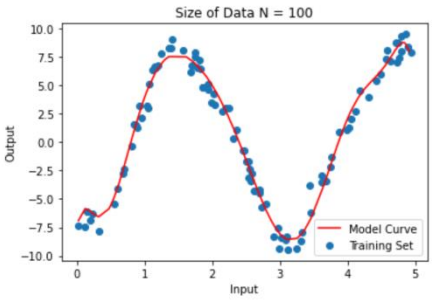
Using Maximum Likelihood Estimation, we wrote a function which takes in the inputs x, y, M(order), and λ(regularization parameter). Φ is made using powers of x till M, and their coefficients (w) are found by minimizing the error between actual values and values predicted by our model.

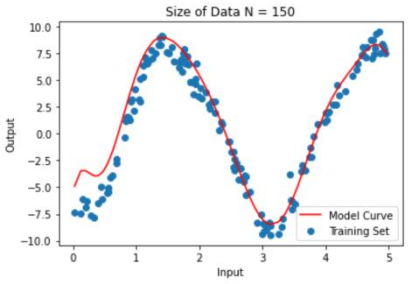
### Data Size

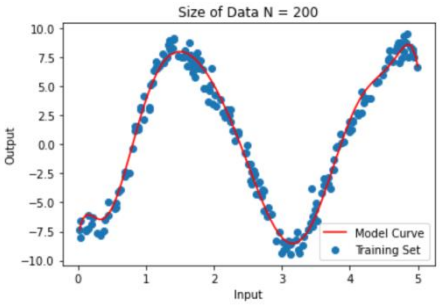


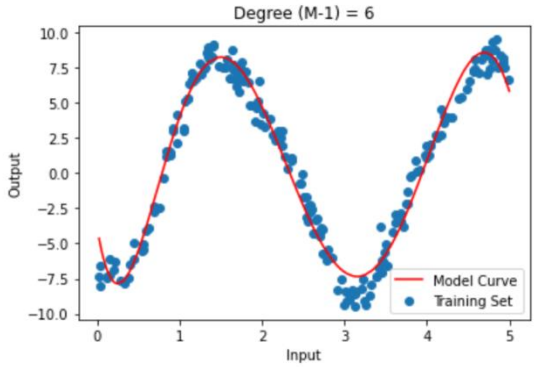



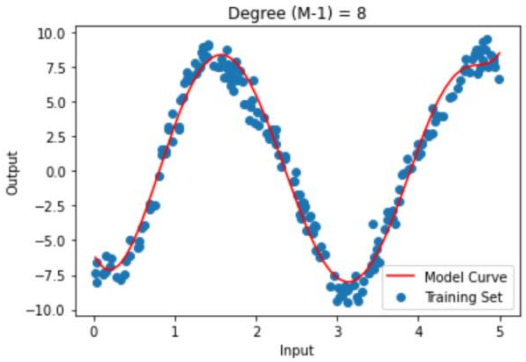



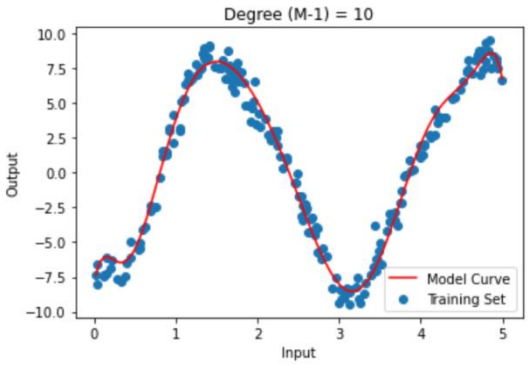



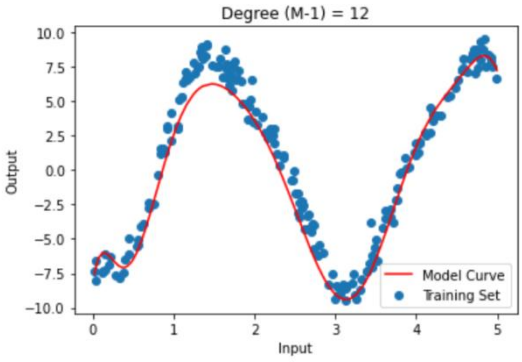

As can be seen from the graphs above, we've taken fractions of the input data in increasing order and have plotted minimum error function. As the size of the training dataset increases, the curves become more refined and smoothens out. This is expected since with more points the curve should fit better.

**Degree of Polynomial**



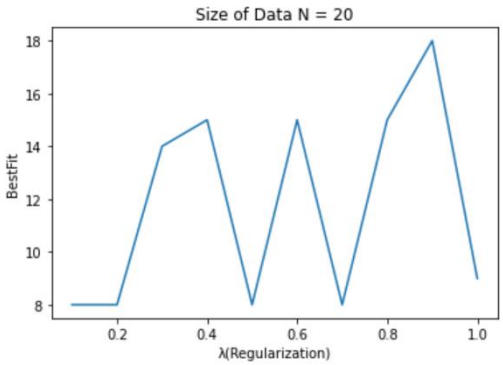



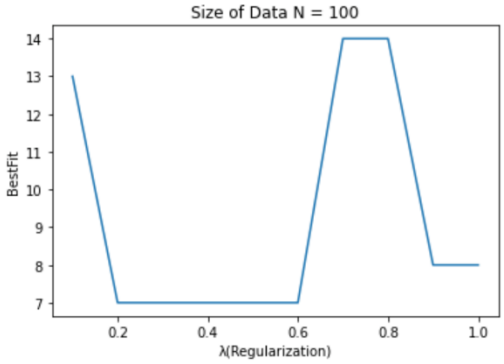



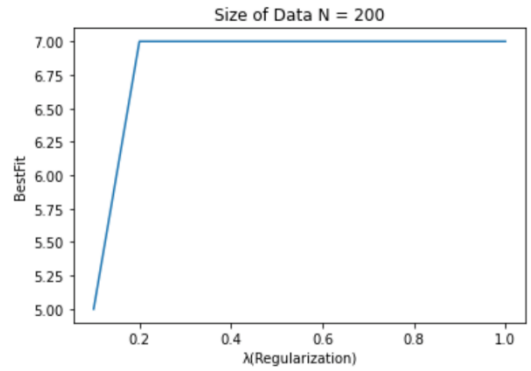




As the degree of the polynomial increases, we expect there to be overfitting since the polynomial will pass through more points to reduce the error. This will lead to the graph becoming less smooth as can be seen from the above plots. We can see how at the two extremes of the graph the curve starts becoming zig-zag and less smooth as M increases.

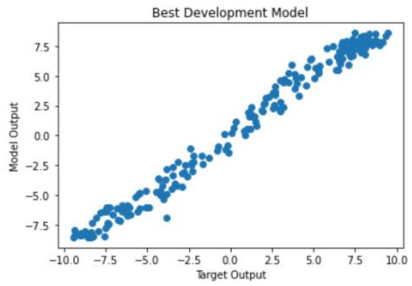
**Regularization Parameter**



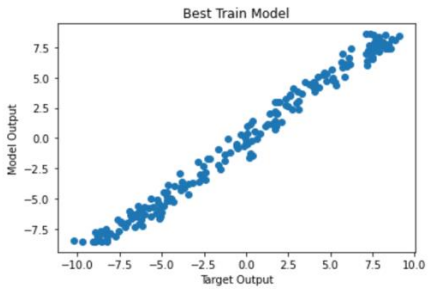






We have written a best fit function which return the polynomial degree which gives minimum error. Now when we use the entire training data set, the degree of bestfit increases with regularization parameter till a certain point after which it remains constant at 7. As we decrease the size of the training data set, we can see that the fluctuations of bestfit degree increase.

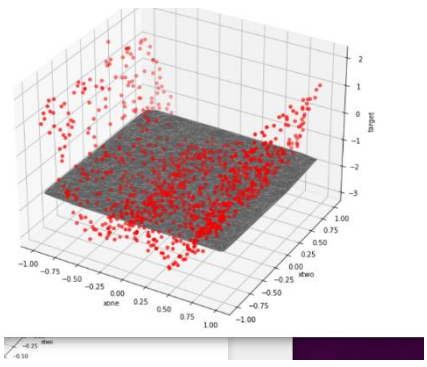



Developmet Error = 129.544

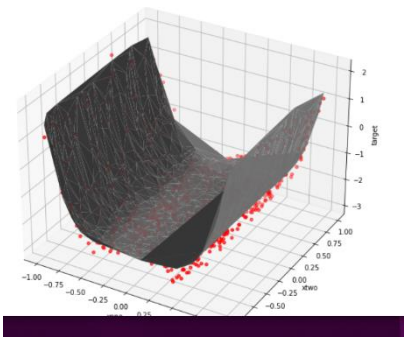



Train Error = 93.345

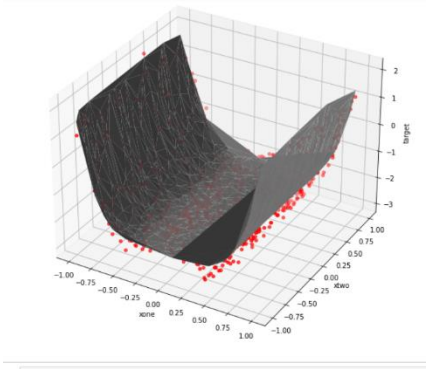
Similarly, for the two-dimensional data:



Degree 2

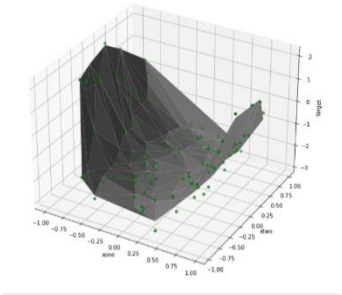


Degree 7

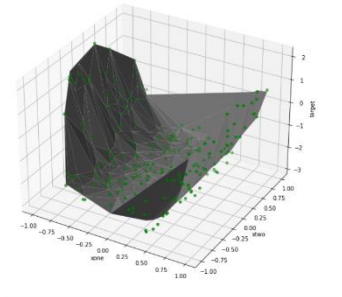


Degree 9

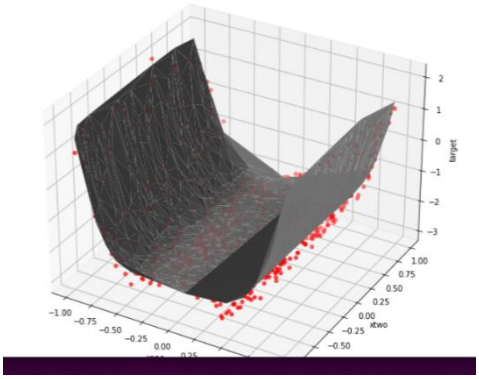
Taking the entire training data set, we can see how increasing the degree of polynomial also increases the fit of the surface. Degree 2 is almost a flat surface, but degree 9 fits to the given points very well.



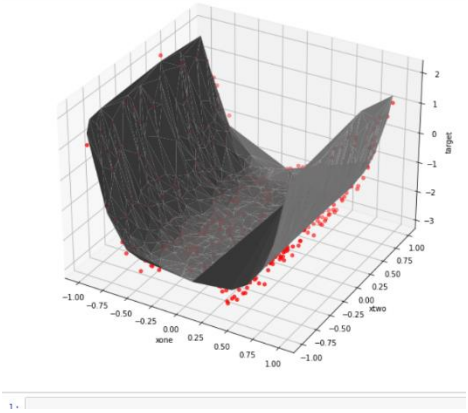

10%



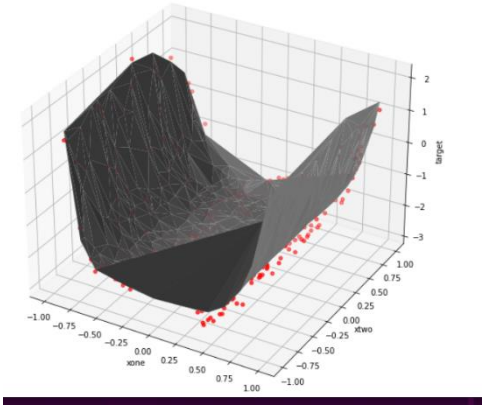

20%




50%

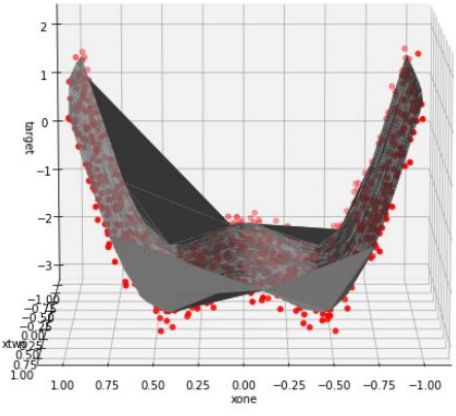


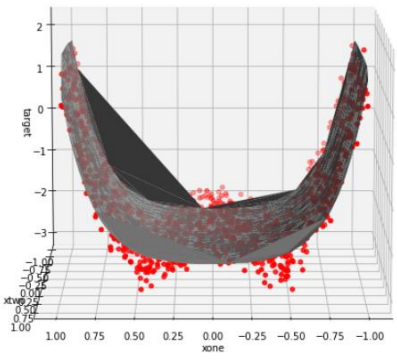

70%



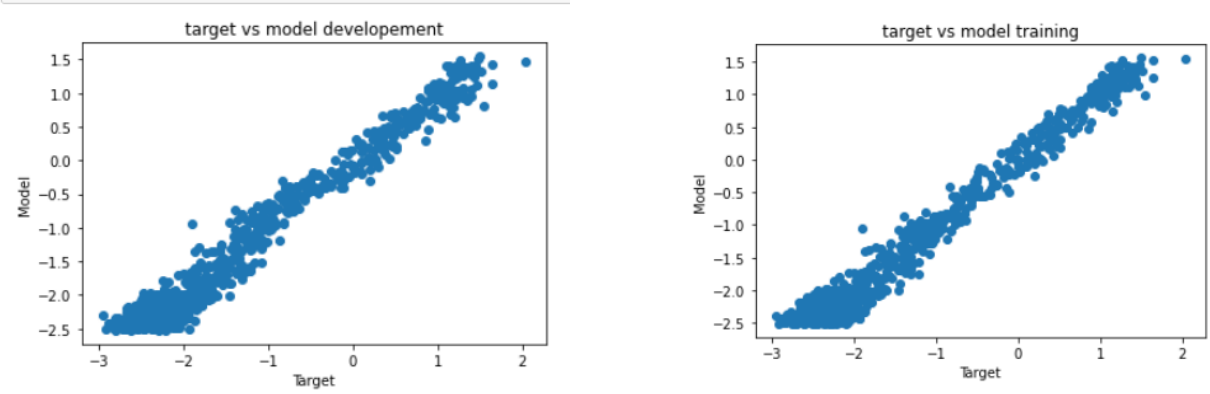

100%

We can see the effect of training data set size in the above graphs. The graph becomes much more smoother as the size increses, and the irregularities on the surface also decrease in size.



Here in the left figure we haven't used any regularization parameters and we get a 'w' shape. In the right we have used a parameter of 7, and the curve turns out to be smoother. Below is the scatter plot for target and model outputs.




The total error value of both training and developement data has been calculated below. For weighted error we would need to divide this value by the total number of poitns which is 1000.

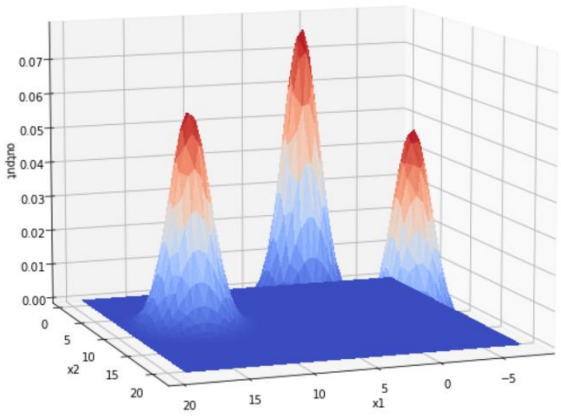
```
In [45]: sum(np.square(t-final))
Out[45]: 36.75138070073222

In [46]: sum(np.square(t-finald))
Out[46]: 43.711773619996784
```

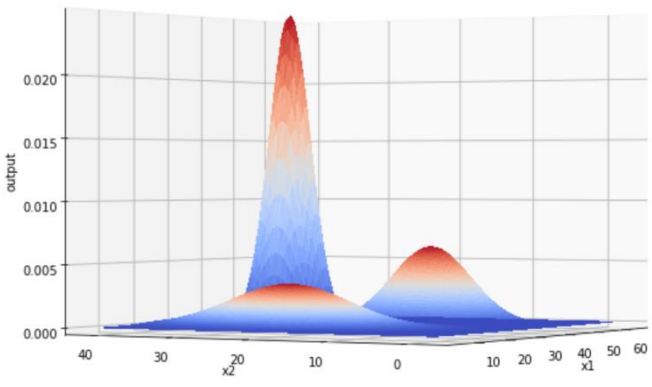
# Bayesian Classifier

1. Bayes with same covariance: found covariance of entire training data set regardless class
2. Bayes with different covariance: found covariance of each given class
3. Naïve Bayes with c=sigma^2I: found covariance of entire data set and made all entries except the diagonal 0. Took average of diagonal values as sigma^2
4. Naïve Bayes with C same: found covariance of entire data set and made non-diagonal entries 0
5. Naïve Bayes with C different: found covariance of individual classes and made non-diagonal entries 0

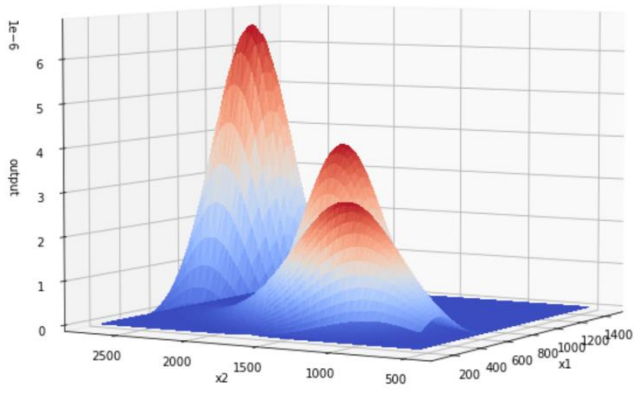
**Probability Density Functions (Gaussians)**




Linear



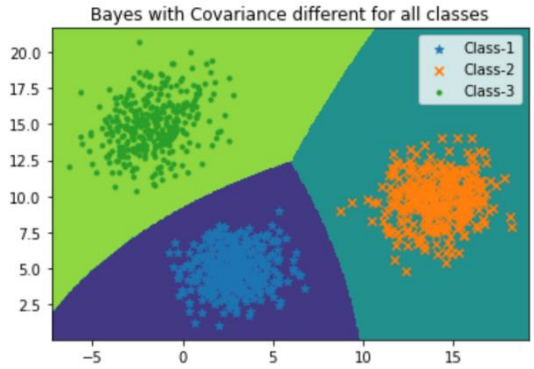

Non-Linear

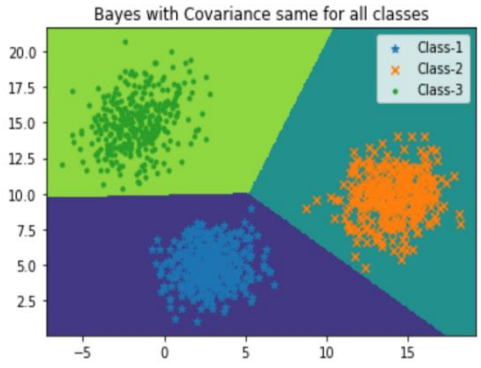


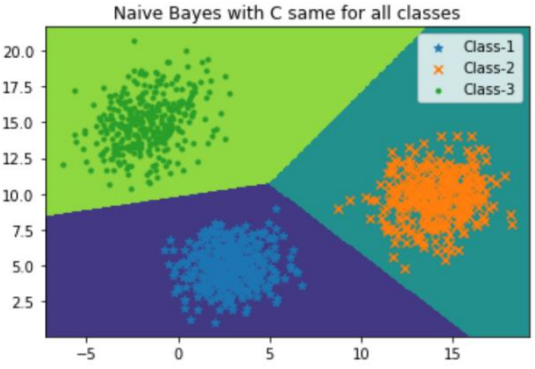

Real

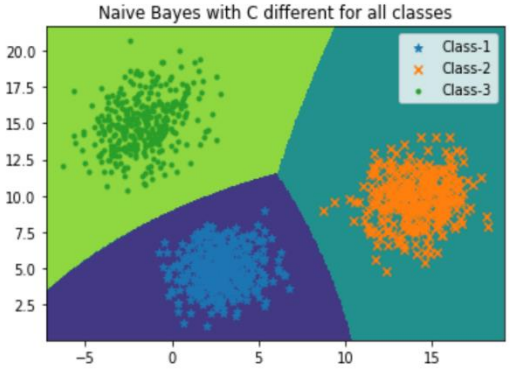
**Decision Boundaries**

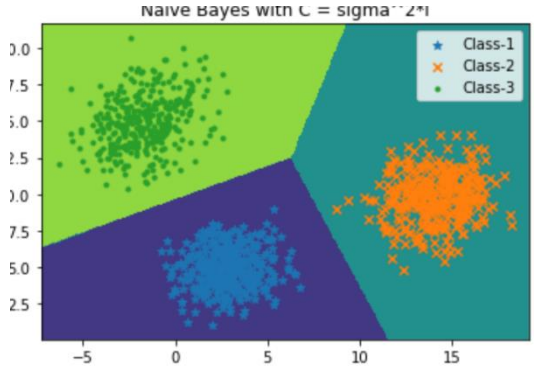
Linear Data



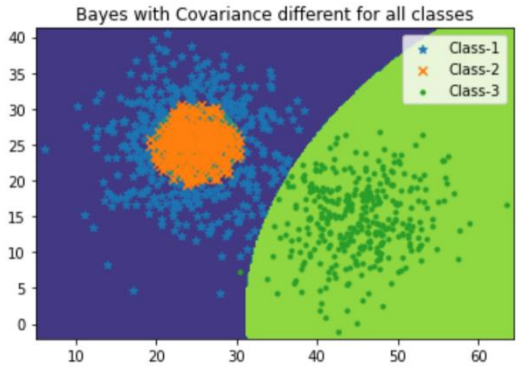



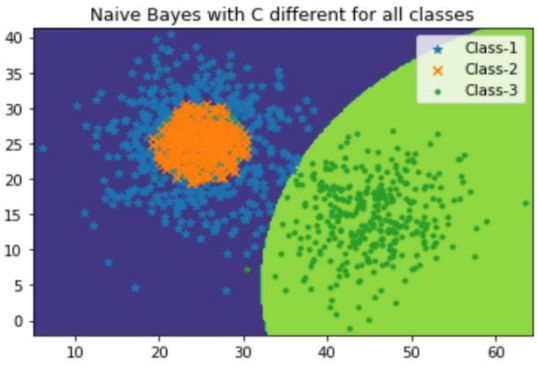



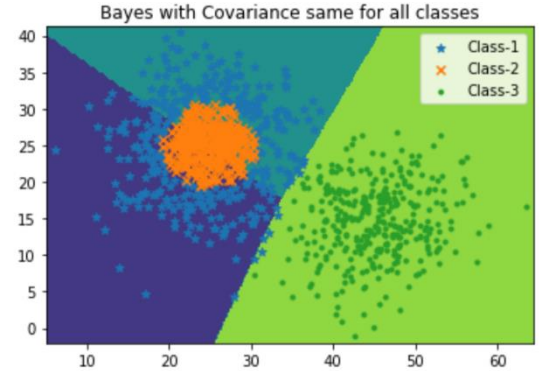



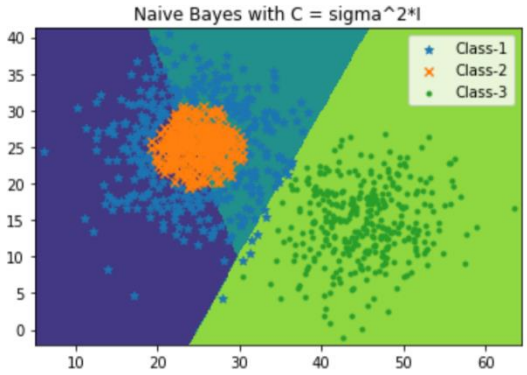



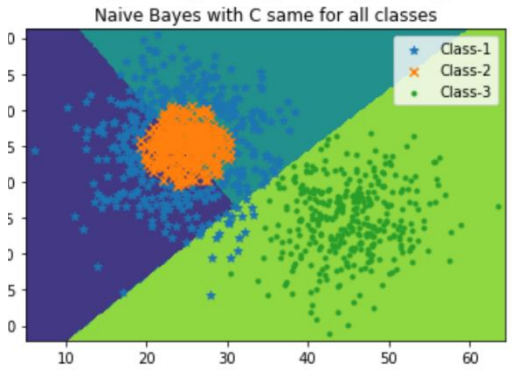
Non-Linear Data



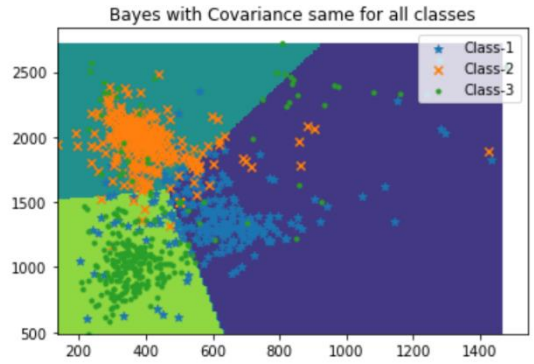



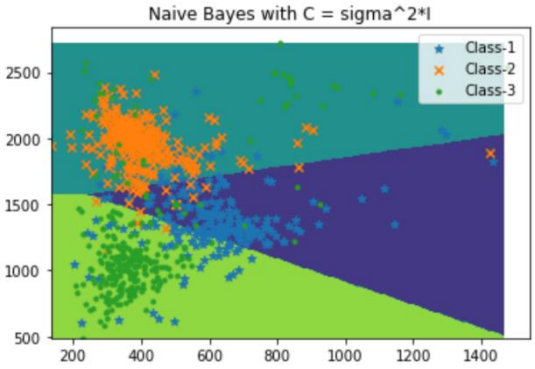



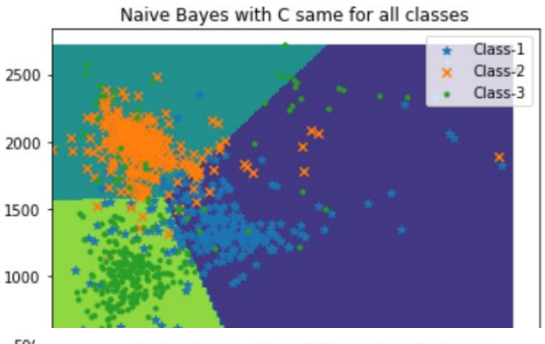



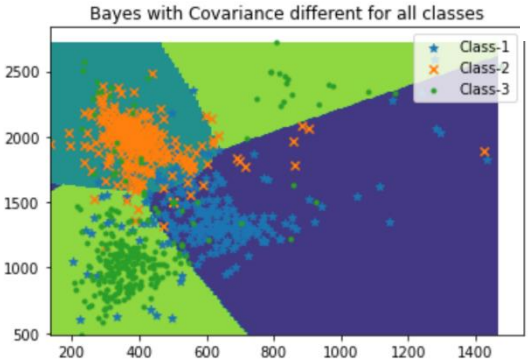



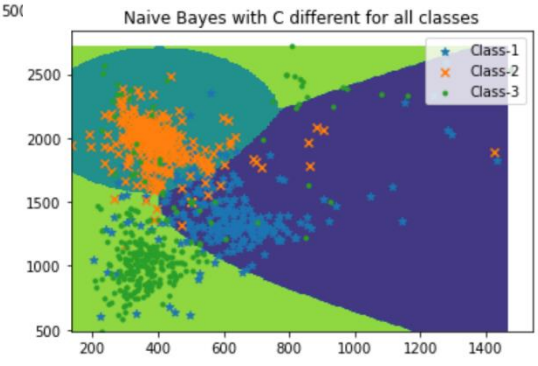
Real Data



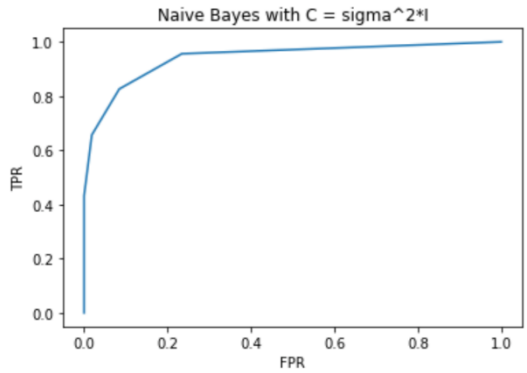



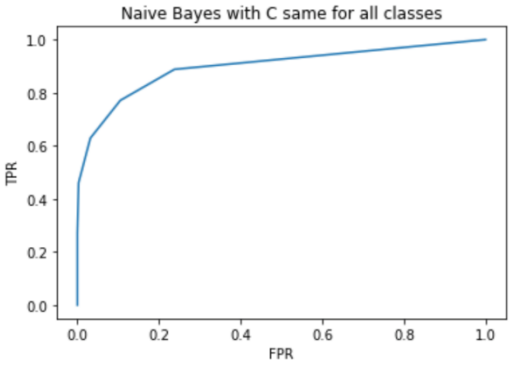



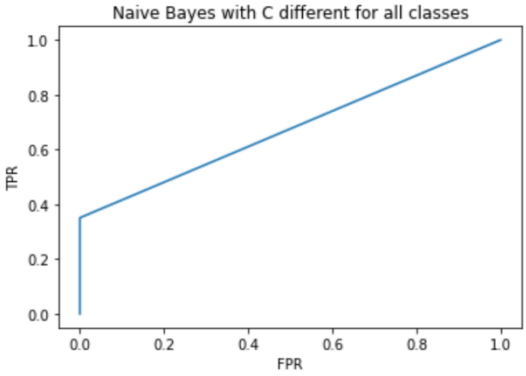



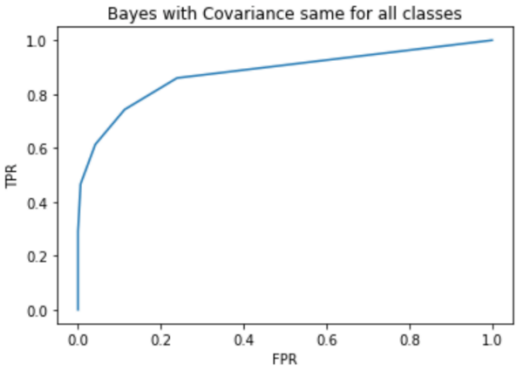



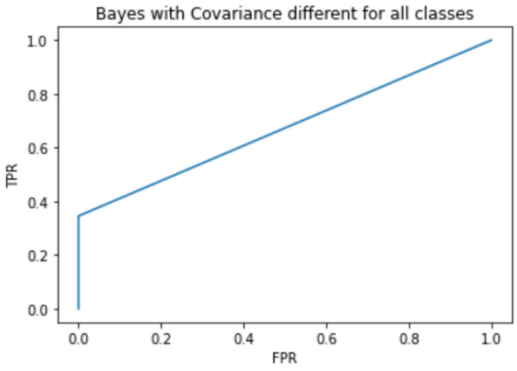

**ROC Curves**


Naive Bayes with C = sigma^2*I
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

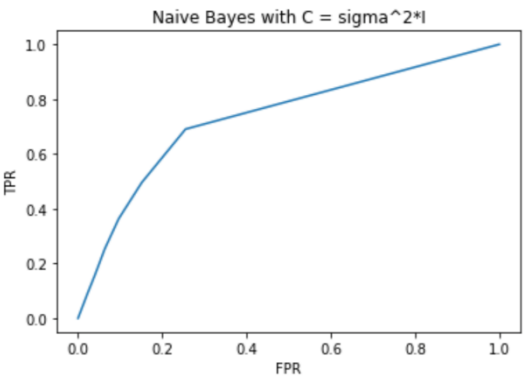

Naive Bayes with C same for all classes
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

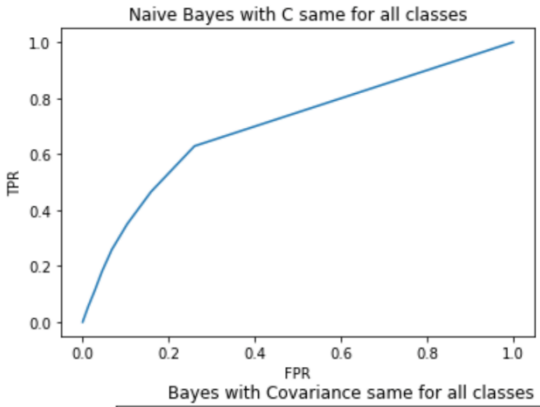

Naive Bayes with C different for all classes
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

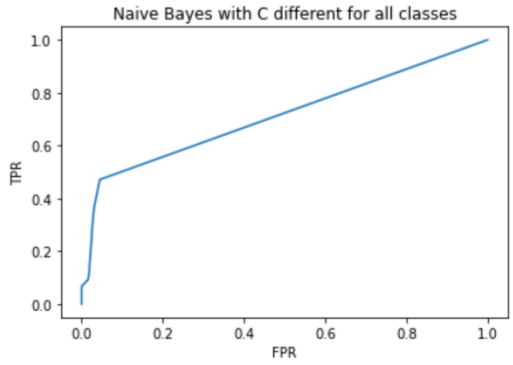

Bayes with Covariance same for all classes
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

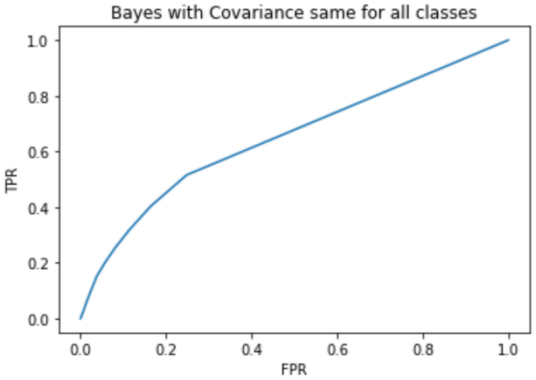

Bayes with Covariance different for all classes
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0

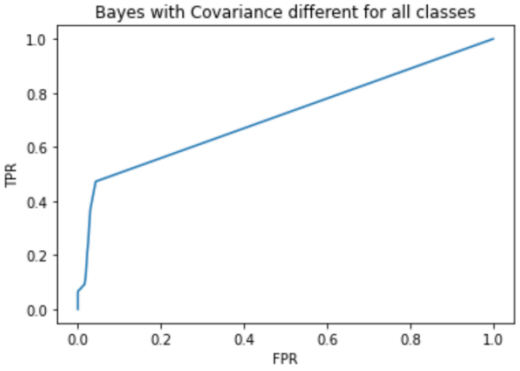
Linear Data



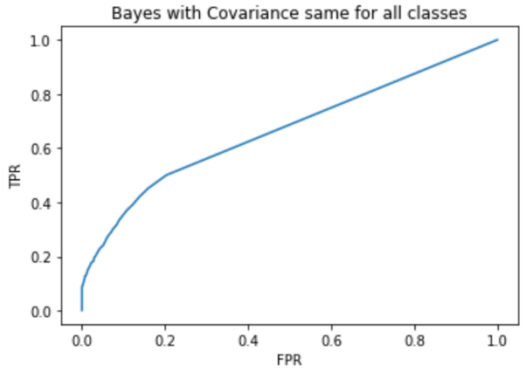



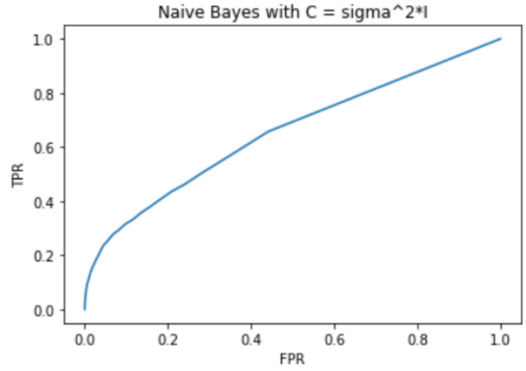



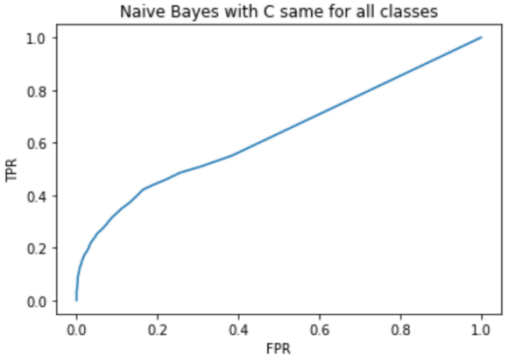



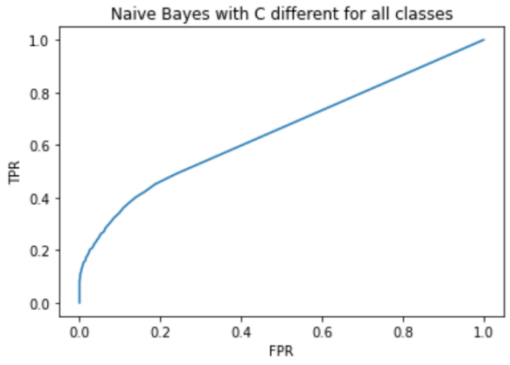

Non-Linear Data

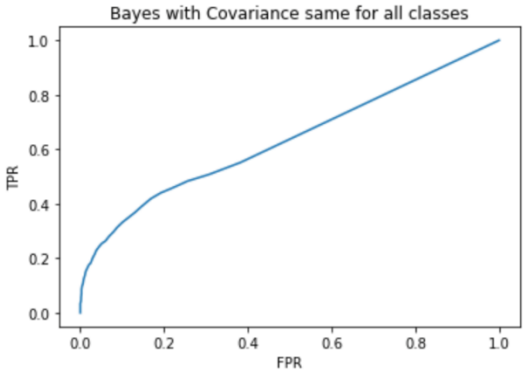



## Real Data




## Confusion Matrix